VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ROZPOZNÁNÍ TYPU VOZIDLA Z DOHLEDOVÉ KAMERY

FINE-GRAINED VEHICLE RECOGNITION FROM TRAFFIC SURVEILLANCE CAMERA

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **PAVEL MENCNER**
AUTHOR

VEDOUCÍ PRÁCE **Ing. JAKUB SOCHOR,**
SUPERVISOR

BRNO 2018

Zadání bakalářské práce/18077/2017/xmencn00

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2017/2018

Zadání bakalářské práce

Řešitel: **Mencner Pavel**
Obor: Informační technologie
Téma: **Rozpoznání typu vozidla z dohledové kamery**
Fine-Grained Vehicle Recognition from Traffic Surveillance Camera
Kategorie: Zpracování obrazu

Pokyny:

1. Nastudujte algoritmy pro detekci vozidla ve videu.
2. Nastudujte algoritmy použitelné pro rozpoznání modelu vozidla.
3. Navrhněte systém pro rozpoznání typu vozidla s využitím dohledové kamery.
4. Navržený systém implementujte.
5. Obstarejte vhodnou datovou sadu pro vyhodnocení přesnosti systému a proveďte toto vyhodnocení.
6. Vytvořte video pro prezentaci vaší práce.

Literatura:

- Dle pokynů vedoucího.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Sochor Jakub, Ing.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 16. května 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
L.S. 612 66 Brno, Božetěchova 2

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Cílem této práce je detekce vozidel v obraze z dopravní dohledové kamery a jemná klasifikace jejich typu (výrobce a model). V práci je implementována normalizační metoda Unpack, která slouží pro transformaci obrazu vozidla do jeho zdánlivé rovinné reprezentace, za účelem zvýšení úspěšnosti klasifikátoru. Metoda Unpack využívá pro normalizaci 3D bounding box vozidla, který je v testovací fázi sestaven z informací o kontuře a směru k úběžníkům vozidla. Součástí práce je srovnání přesnosti metody přímé a Unpack klasifikace. Řešení se skládá z více na sebe navazujících částí, které využívají konvolučních neuronových sítí. Tyto části jsou: detekce vozidel v obraze, odhad směru k úběžníkům scény řešený jako klasifikační úloha, detekce kontury vozidel s využitím konvoluční Encoder-Decoder sítě a jemná klasifikace typu vozidel. Pomocí klasifikace s využitím metody Unpack bylo dosaženo zvýšení přesnosti systému o 2% proti přímé klasifikaci, dosahujíc výsledné úspěšnosti 86%. Výsledkem práce je systém jemné klasifikace typu vozidel pracující se záznamem z dohledové kamery bez omezení pozorovacích úhlů.

Abstract

The aim of this thesis is image based detection of vehicles from traffic surveillance camera and fine-grained vehicle type recognition (manufacturer and model). In the thesis the Unpack normalization method is implemented which transforms the vehicle image into its apparent flat representation in order to increase the classifier's success rate. The Unpack method make use of 3D bounding box of the vehicle. This bounding box is constructed during test period using the information of vehicle contour and direction toward vanishing points. The thesis involve accuracy comparison between direct and Unpack classification methods. The proposed solution is based on several related parts that benefit from convolutional neural networks. These parts are: vehicle detection from image data, estimation of the directions towards vanishing points solved as classification task, vehicle contour detection using convolutional Encoder-Decoder network and fine-grained vehicle type classification. Using Unpack based classification the 2% accuracy improvement against direct classification has been achieved, resulting in 86% overall success rate. The outcome of this thesis is fine-grained vehicle classification system that works with traffic surveillance video without any viewpoint limitations.

Klíčová slova

zpracování obrazu, konvoluční neuronové sítě, analýza dopravy, dopravní dohledová kamera, detekce vozidel, jemná klasifikace typu vozidel, odhad kontury objektu, odhad směru k úběžníkům, 3D bounding box, Unpack, Tensorflow, Keras, Python, OpenCV

Keywords

image processing, convolutional neural network, traffic analysis, traffic surveillance camera, vehicle detection, fine-grained vehicle type recognition, object contour estimation, vanishing point estimation, 3D bounding box, Tensorflow, Keras, Python, OpenCV

Citace

MENCNER, Pavel. *Rozpoznání typu vozidla z dohledové kamery.* Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Jakub Sochor,

Rozpoznání typu vozidla z dohledové kamery

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jakuba Sochora. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Pavel Mencner
16. května 2018

Poděkování

Tímto bych chtěl poděkovat panu Ing. Jakubu Sochorovi za konzultace a vedení této práce. Taktéž bych chtěl poděkovat své přítelkyni Lauře za podporu a shovívavost.

Obsah

Kapitola 1

Úvod

Dohledové kamery se staly součástí každodenního života. Jejich přítomnost umožňuje mít neustálý dohled nad důležitými aspekty naší společnosti a i přes možné diskutabilní důsledky na naše soukromí, je jejich přínos pro každodenní bezpečnost nepopiratelný. Jednou z oblastí, ve které nachází dohledové kamery široké uplatnění, je monitorování dopravní situace podstatných dopravních uzlů. Poskytují nám možnost statistického a krizového monitorování. S rostoucím množstvím získaných obrazových dat je však kladen stále větší důraz na jejich automatické zpracování. Na scénu tak nastupuje počítačové zpracování obrazu, které je neustále se vyvíjející oblastí a kterému již několik let dominují, co se oblíbenosti, úspěšnosti a pozornosti výzkumníků týče, konvoluční neuronové sítě.

Tato práce se věnuje jednomu ze základních problémů zpracování obrazu v analýze dopravy, detekci vozidel a následné klasifikaci výrobce a modelu osobního vozidla. Již tato základní informace získaná z obrazu, nám může poskytnout mnoho informací o statistice vytíženosti dopravní komunikace, složení vozového parku nebo přítomnosti dopravních anomálií jako je např. dopravní zácpa. Velkou výhodou takového řešení je, že může být nasazeno do stávající informační infrastruktury bez nutnosti zásahu do vozovky, investic do hardwarového vybavení a instalace kvalitních dopravních kamer.

Analýzou existujících řešení byl zjištěn důraz na normalizaci vstupních dat pro zlepšení výsledků klasifikace. V kontextu práce se jedná o obecně libovolný úhel pohledu dohledové kamery. Vlivem této skutečnosti je dán důraz na realizaci odhadu 3D bounding boxu vozidla bez nutnosti manuální kalibrace scény a následná normalizace obrazových dat pomocí metody Unpack. Následně je zkoumán vliv aplikace této normalizační metody na úspěšnost klasifikace, přesnost dílčích částí systému, výsledná úspěšnost systému a různé typy architektur konvolučních neuronových sítí.

Kapitola 2

Existující řešení klasifikace typu vozidel

Cílem této práce je jemná klasifikace typu vozidel. Pod pojmem jemná klasifikace se rozumí klasifikace výrobce, modelu a typu karoserie vozidla včetně rozlišení modelové řady (např. Škoda Octavia II kombi). Problematikou jemné klasifikace typu vozidel se již zabývalo mnoho prací. Na rozdíl od obecné klasifikace objektů se v případě klasifikace vozidel lze spoléhat na přítomnost určitých společných prvků jako je pozice registrační značky nebo světel vozidla. Přesto tento úkol zůstává výzvou především kvůli variabilitě pozorovacích úhlů a naopak nízké variabilitě mezi jednotlivými třídami klasifikace. V následující kapitole budou rozebrány způsoby řešení podle přístupu, využitých metod a vyplývajících omezení řešení.

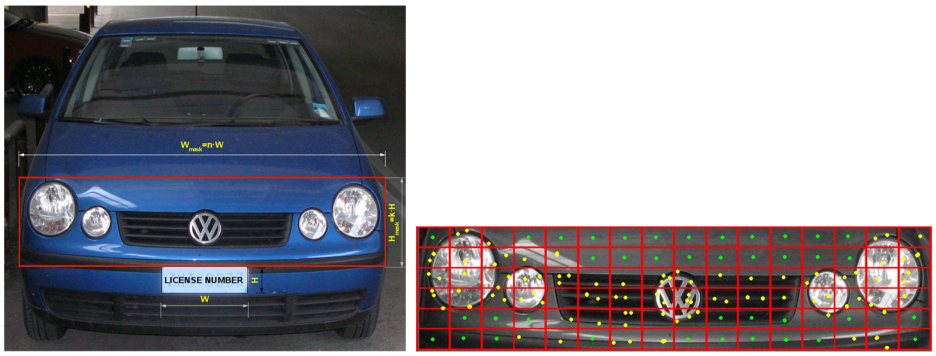
2.1 Metody založené na detekci registrační značky

Práce *Multi-class Vehicle Type Recognition System* [3] pracuje s jemnou klasifikací typu pomocí orientovaných bodů kontury vozidla. Metoda předpokládá přesnou znalost pozici rohů registrační značky vozidla $\{A, B, C, D\}$, které jsou využity pro normalizaci vstupních dat klasifikace pomocí geometrických transformací obrazu pro získání požadované referenční pozice registrační značky $\{A', B', C', D'\}$, jak je znázorněno na obrázku 2.1. Tím je dosaženo normalizace pozice jednotlivých klíčových částí vozidla jako maska nebo světla. Pro získání kontury obrazu jsou využity Sobelovy operátory, které pracují s konvolučním jádrem. Pomocí tohoto jádra aproximují derivaci jasové funkce v obraze a detekují tak hrany. Klasifikace probíhá algoritmem *k-nejbližších sousedů (k-NN)*. Avizovaná úspěšnost klasifikace dosahuje 90%. Limitací této metody je její omezení na přední pohled vozidla a nutnost znalosti pozice registrační značky.

Práce *Vehicle model recognition from frontal view image measurements* [16] také využívá snímky vozidel pořízené z předního pohledu a pracuje s detekcí pozice registrační značky vozidla, ovšem odlišným způsobem. Na základě pozice registrační značky je definován region přední masky vozidla. Maska je následně segmentována pro získání regionu loga výrobce, které se typicky nachází uprostřed masky. Segmentované logo je klasifikováno pomocí pravděpodobnostní neuronové sítě za účelem identifikace výrobce vozidla. Klasifikace modelu vozidla probíhá v dalším kroku. Podle zjištěného výrobce je vybrána množina příslušných modelů a klasifikace probíhá pomocí algoritmu *SIFT (Scale Invariant Feature Transform)*, který pracuje s detekcí klíčových bodů snímku (viz obrázek 2.2). Detekované klíčové body

Obrázek 2.1: Ukázka normalizace obrazu vozidla pomocí znalosti pozice registrační značky z práce *Multi-class Vehicle Type Recognition System* [3]. **Vlevo:** vstupní obraz a pozice registrační značky, **vpravo:** požadovaný normalizovaný obraz pomocí geometrických transformací obrazu.

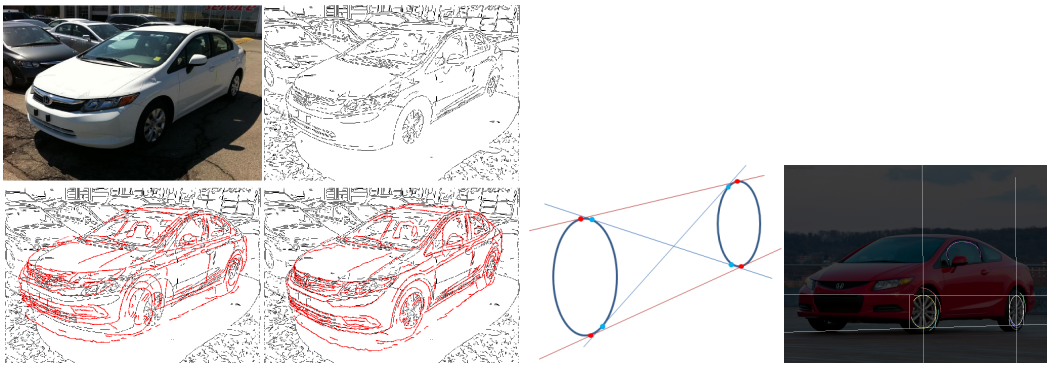
jsou použity jako vstup do další pravděpodobnostní neuronové sítě. Určení výrobce vozidla dosahuje úspěšnosti 85%, určení modelu vozidla 54%.



Obrázek 2.2: Ukázka z práce *Vehicle model recognition from frontal view image measurements* [16]. **Vlevo:** Region přední masky určený pomocí pozice registrační značky. **Vpravo:** Detekce klíčových bodů pomocí algoritmu SIFT (žlutě skutečné klíčové body, zeleně nulové body).

2.2 Metody s využitím 3D modelů

Práce *Car Make and Model Recognition using 3D Curve Alignment* [17] využívá pro klasifikaci třídy vozidla parciální 3D modely tvořené křivkami, které jsou získány automaticky z obrazových dat. Každý model vozidla je reprezentován sadou referenčních snímků pořízených z různých úhlů záběru, které jsou převedeny na 3D modely pomocí metody detekce hran. Metoda si neklade žádné omezení na pozorovací úhly klasifikovaného záběru. Toho je docíleno odhadem pozorovacího úhlu založeného na detekci elipsy kol vozidla a zkoumáním jejich vzájemného relativního umístění. Na základě odhadu pozorovacího úhlu jsou zvoleny odpovídající referenční parciální 3D modely pro proces klasifikace. Klasifikace probíhá na základě nalezení shody referenčního a klasifikovaného 3D modelu s využitím procesu normalizace. Normalizace probíhá pomocí transformace pro minimalizaci chyby projekce sobě náležících křivek (viz obrázek 2.3). Pro tuto normalizaci je využita informace o odhadu pohledu scény obou modelů. Metoda dosahuje průměrné úspěšnosti 80%.



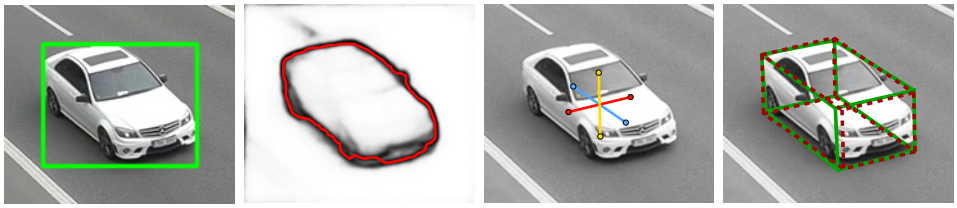
Obrázek 2.3: *Car Make and Model Recognition using 3D Curve Alignment* [17]. **Vlevo:** proces klasifikace – vstupní obrázek, detekované hrany (3D model), přímá shoda s referenčním modelem, shoda po normalizaci dat pomocí odhadu pozorovacího úhlu. **Vpravo:** detekce elipsy kol vozidla pro odhad pozorovacích úhlů.

Práce *3D Object Representations for Fine-Grained Categorization* [14] pracuje s kompletními 3D CAD modely reprezentujícími jednotlivé modely vozidel. Tyto 3D modely jsou využity pro generování syntetické datové sady, která obsahuje kartézský součin diskrétních pozorovacích úhlů (36 úhlů, 4 výšky pohledu, 10 různých pozadí) a kategorií vozidel (sedan, SUV, coupe, kabriolet, pick-up, hatchback, kombi). Výsledkem je 1008 jednotlivých HOG (*Histogram of oriented gradients*) klasifikátorů z nichž se bere v potaz N nejlepších výsledků (určujících pozorovací úhel a kategorii) pro další zpracování. Pro určení jemné kategorie vozidla se používá algoritmus SVM (*Support vector machine*) z oblasti strojového učení a to tak, že se již hledá shoda konkrétního modelu na základě dříve zjištěných informací o úhlu pohledu a kategorii. Hlavním omezením metody je nutnost přítomnosti 3D modelu každého klasifikovaného vozidla.

2.3 Metody s využitím konvolučních neuronových sítí

Práce *Fine-Grained Vehicle Model Recognition Using A Coarse-to-Fine Convolutional Neural Network Architecture* [6] využívá konvoluční neuronové sítě pro identifikaci jedinečných částí vozidel a jejich následnou klasifikaci. Toho je docíleno analýzou výstupu konvoluční vrstvy – příznakové mapy. Příznaková mapa identifikuje světlomety a středovou masku vozidla. Práce se zaměřuje pouze na jeden konkrétní úhel pohledu a dosahuje s tímto specifikem úspěšnosti až 98%.

Práce *BoxCars: Improving Fine-Grained Recognition of Vehicles using 3D Bounding Boxes in Traffic Surveillance* [21] využívá konvoluční neuronové sítě pro klasifikaci typu vozidla s využitím metody *Unpack*. Podstatou této metody je získání normalizovaného obrazu vozidla pro účely klasifikace a tím docílení značného zvýšení úspěšnosti (až o 12% proti metodě bez využití metody *Unpack*). Realizace metody *Unpack* je dosaženo pomocí odhadu 3D bounding boxu vozidla, který je konstruován z informací o kontuře vozidla a směrech k úběžníkům od vozidla (viz obrázek 2.4). Samotný odhad směru k úběžníkům je řešen jako klasifikační úloha pomocí konvolučních neuronových sítí. Metoda si neklade žádné omezení na vstupní data, pracuje se snímky pořízenými z libovolného úhlu záběru a dosahuje úspěšnosti až 93%.



Obrázek 2.4: *BoxCars: Improving Fine-Grained Recognition of Vehicles using 3D Bounding Boxes in Traffic Surveillance* [21]. **Zleva postupně:** detekce regionu vozidla, odhad kontury, odhad směru k úběžníkům a 3D bounding box vozidla (zeleně odhad, červeně skutečný 3D bounding box).

2.4 Vyhodnocení metod

Jemnou klasifikací typu vozidel se zabývá mnoho prací, které lze rozdělit do několika skupin podle použité metodologie. Práce v kapitole 2.1 využívají pozici registrační značky jako referenční informaci o pozici vozidla a kladou si specifické omezení na pozorovací úhly záběru, které nemusí být možné dohledovou kamerou splnit. Práce v kapitole 2.2 využívají různým způsobem referenční 3D modely pro docílení libovolného úhlu pořízení záběru. Mají však specifické požadavky na referenční snímky pro konstrukci 3D modelu nebo přímo nutnost přítomnosti 3D CAD modelu každého modelu vozidla. Práce v kapitole 2.3 využívají pro klasifikaci konvoluční neuronové sítě v kombinaci s různými technikami identifikace zájmových regionů vozidla. Tyto metody dosahují výborných výsledků a v kombinaci s možnosti řešení pro obecné pozorovací podmínky a dobrou dostupností referenčních dat (oproti 3D CAD modelům) tvoří zajímavou metodu řešení.

Kapitola 3

Zpracování obrazu pomocí konvolučních neuronových sítí

Tato kapitola se zabývá problematikou zpracování obrazových informací pomocí konvolučních neuronových sítí. Mezi základními úkoly zpracování obrazu je možné zařadit detekci objektu a klasifikaci jeho typu. Tato činnost je pro člověka triviální a zvládá ji bez sebemenších potíží. Přesto se v oblasti počítačového zpracování obrazu stále jedná o netriviální úkol, kterému se věnuje řada výzkumníků, pořádají se v něm soutěže na světové úrovni (*ImageNet Large Scale Visual Recognition Challenge*[1]) a díky tomu dochází v této oblasti k neustálému zlepšování výsledků. Těmto soutěžím a výzkumům co do výsledků úspěšnosti již několik let v mnoha oblastech úspěšně dominují právě zmíněné konvoluční neuronové sítě [20]. Náročnost úkolu klasifikace je dána principem reprezentace obrazových informací, jakožto matice pixelů o různých intenzitách a teoreticky nekonečnou množinou variací reprezentující danou třídu, které se mohou lišit v mnoha aspektech [13]. Možnosti variability, které jsou znázorněny na obrázku 3.1, mohou být následující:

- Úhel pohledu na objekt
- Velikost (rozměry objektu či vzdálenost pořízení snímku)
- Deformace objektu
- Překrývání s jinými objekty
- Světelné podmínky scény (jas, stíny)
- Variabilita pozadí
- Variace uvnitř samotné třídy

3.1 Umělá inteligence a strojové učení

Pojem umělá inteligence je obecně definován jako digitální technologie, která je schopna vykonávat úkoly jejichž řešení zdánlivě vyžaduje inteligentní chování [1]. Může se jednat o úkoly typu identifikace nevyžádané pošty v e-mailové komunikace, klasifikace typu objektu, rozpoznání řeči, autonomní řízení vozidla a nespočet jiných aplikací.

Strojové učení lze chápat jako podmnožinu oboru umělé inteligence a tímto pojmem se vyznačuje systém schopný zlepšování vlastního chování v řešení určité úlohy. Proces trénování lze popsat jako snahu generalizovat určité příznaky sledováním vzorků z trénovací

[1] http://www.image-net.org/challenges/LSVRC/

Obrázek 3.1: Ukázka možnosti diverzity snímků reprezentující stejnou třídu objektů, v tomto případě vozidlo Škoda Octavia.

množiny [5]. Tento přístup se osvědčil v mnoha odvětvích, kde se kvůli stále komplikovanějším úlohám ukázal jako atraktivní alternativa k ručnímu programování příznaků a metod (explicitní definice všech entit, algoritmů a parametrů). Konvoluční neuronové sítě jsou právě jedním z prostředků realizace systémů využívajících strojové učení a umělé inteligence především v oblasti zpracování obrazu. Mezi aplikace strojového učení patří:

- **Klasifikace** - Určení třídy příslušnosti objektu
- **Regrese** - Zjištění vztahu mezi proměnnými (odhad výstupu na základě vstupu)
- **Shlukování** - Seskupení objektů stejné skupiny

3.1.1 Učení s učitelem

Jedním ze základních způsobů dělení metod strojového učení je podle přístupu k procesu učení, konkrétně přítomnost určité zpětnovazební informace ovlivňující průběh učení.

Základem učení s učitelem jsou anotovaná vstupní data. Anotací se v tomto případě rozumí informace představující požadovaný výstup pro daná vstupní data. Tato data jsou rozdělena do dvou disjunktních množin představující trénovací a testovací vzorky [7].

Cílem učení je vytvoření co nejpřesnějšího modelu jeho iterativním zlepšováním na základě analýzy trénovací vstupní množiny. Testovací množina obsahuje vzorky stejných tříd jako množina trénovací a slouží k ověření úspěšnosti trénovaného modelu. Poskytuje tak zpětnou vazbu pro další iterace učení.

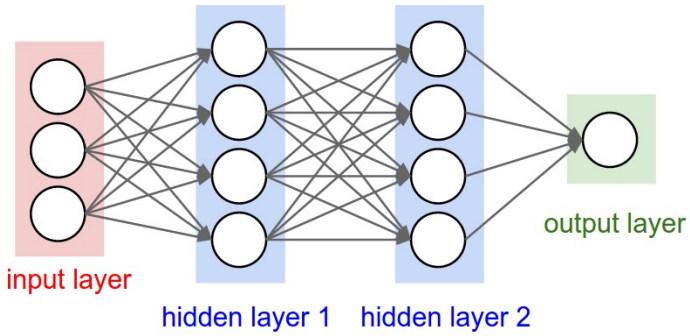
Formálně řečeno se trénovací i testovací množina skládá z N (rozdílné pro trénovací a testovací množinu) uspořádaných dvojic (x_i, y_i), kde x_i je množina vstupních informací a y_i je anotace resp. požadovaný výstup klasifikátoru. Například x_i může představovat obrázek s ručně psanou číslicí a y_i textovou informaci, o kterou číslici se jedná.

3.2 Konvoluční neuronové sítě

Model umělé neuronové sítě je odvozen z principu fungování biologického neuronu v mozku. Tato souvislost je však pro téma okrajová a dále je pod pojmem neuron vždy myšlen neuron z oblasti umělé inteligence. Neuronová síť se skládá z množiny vzájemně propojených

neuronů. Tyto neurony přijímají vstupy neuronové sítě nebo výstup jiného neuronu. Funkcí každého neuronu je transformace vektoru vstupů na výstup pomocí specifické přenosové funkce [11].

Neurony neuronové sítě jsou shlukovány a tvoří tak vrstvy. Jednotlivé vrstvy pak plní svou specifickou roli v architektuře celé neuronové sítě. Jednotlivé vrstvy jsou propojeny, a to tak, že výstup předchozí vrstvy tvoří vstup vrstvy následující. Vrstvy lze rozdělit podle jejich umístění na vstupní, výstupní (tvoří vstup/výstup sítě) a skryté (vnitřní vrstvy sítě) [12]. Jednoduchá síť, která je tvořena vstupní, dvěma skrytými a výstupní vrstvou je znázorněna na obrázku 3.2. Konkrétní typy vrstev a jejich funkce v architektuře jsou podrobněji rozebrány v kapitole 3.2.2.



Obrázek 3.2: Jednoduchá neuronová síť obsahující vstupní (input), dvě skryté (hidden) a výstupní (output) vrstvu. Zdroj obrázku [12].

Pro neuronovou síť s více než jednou skrytou vrstvou se užívá pojem hluboká neuronová síť a s tím spojené hluboké učení. Vzhledem k tomu, že jednotlivé vrstvy hlubokých neuronových sítí se učí vždy od vrstev předchozích, se v hlubších vrstvách nachází více generalizované a abstraktní příznaky. S rostoucí hloubkou sítě roste její složitost a potenciálně i její úspěšnost. V kombinaci s dostupností výpočetního výkonu nejsou dnes výjimkou sítě s hloubkou v řádu desítek až stovek skrytých vrstev.

Konvoluční neuronové sítě jsou podobné klasickým hlubokým neuronovým sítím. Jsou tvořeny vzájemně propojenými vrstvami neuronů. Liší se však tím, že explicitně předpokládáme vstup v podobě obrazových dat (alternativně se používá vstup v podobě zvukové stopy [4]). Jak název napovídá výraznou roli sehrává přítomnost matematické operace konvoluce. Konvoluční neuronová síť je tedy speciálním případem hluboké neuronové sítě, kde je alespoň v jedné z vrstev použita konvoluce namísto obecné maticové operace.

3.2.1 Učení neuronové sítě

Pro učení neuronové sítě se nejčastěji používá algoritmus zpětného šíření chyby (*backpropagation*). Jeho cílem je úprava vah sítě tak, aby skutečný výstup sítě co nejvíce odpovídal požadovanému výstupu a to pro všechny vstupy z dané trénovací množiny.

Algoritmus zpětného šíření chyby slouží k minimalizaci chybové funkce *Loss*. Na počátku procesu trénování jsou váhy sítě nastaveny typicky na náhodné hodnoty a na vstup sítě je přiveden vstupní vektor (matice) a je proveden průchod sítí pro získání výstupů. Na počátku trénování nebude výstup sítě odpovídat skutečnosti. V té chvíli přichází na řadu funkce *Loss*, která může být definována různě, typicky však pomocí střední kvadratické chyby (*MSE* –

mean square error) definované v rovnici 3.1, kde n je počet trénovacích vstupů iterace, y_i je výstupní hodnota z neuronové sítě a t_i je požadovaná výstupní hodnota [15].

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(t_i - y_i)^2 \tag{3.1}$$

Poté přichází na řadu samotný algoritmus zpětného šíření chyby. Jedná se o gradientní metodu, která iterativně adaptuje každou váhu na základě parciální derivace chybové funkce. Rychlost této adaptace je ovlivněna parametrem learning rate určujícím velikost změny vah vůči gradientu. Vyšší hodnota parametru learning rate znamená rychlejší konvergenci parametrů, pokud je však parametr příliš vysoký, může docházet k oscilaci okolo požadovaného výsledku a nedosažení požadované konvergence.

3.2.2 Vrstvy konvolučních neuronových sítí

Vrstvy jsou základním abstraktním celkem návrhu architektury konvoluční neuronové sítě. Každá vrstva transformuje svůj vstup v podobě matice pomocí specifické aktivační funkce na výstupní matici. Pro návrh architektury konvolučních sítí se využívají 3 základní typy vrstev: konvoluční (convolutional), sdružovací (pooling) a plně propojená (fully connected) vrstva.

Konvoluční vrstva

Konvoluční vrstva tvoří základní stavební kámen každé konvoluční neuronové sítě. Jejím základem je matematická operace konvoluce. Principem konvoluce je aplikace konvolučního jádra K na vstupní obraz I na jeho konkrétním indexu (i, j) a to jako suma násobků hodnot prvků konvolučního jádra a obrazu na sobě odpovídajících indexech [9]. Vztah je znázorněn rovnicí 3.2.

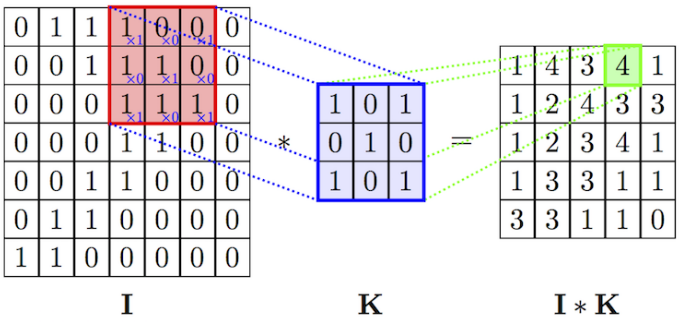
$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \tag{3.2}$$

Konvoluční jádro má parametrizovaný rozměr (např. 5x5x3 značí filtr o rozměru 5x5 pixelů vždy přes všechny 3 kanály barevného obrazu). Aplikace filtru probíhá přes všechny možné pozice a vytváří tak dvourozměrnou aktivační mapu, která reaguje při výskytu určitého vizuálního příznaku. Cílem procesu učení je nastavení vah filtru tak, aby detekoval generalizované příznaky dané třídy [12]. Ve vrstvách nižší úrovně se může jednat o hranu určité orientace nebo barevný vzor, v hlubších vrstvách větší abstraktní celky jako kolo nebo obličej. Princip operace konvoluce je znázorněn na obrázku 3.3.

Sdružovací vrstva

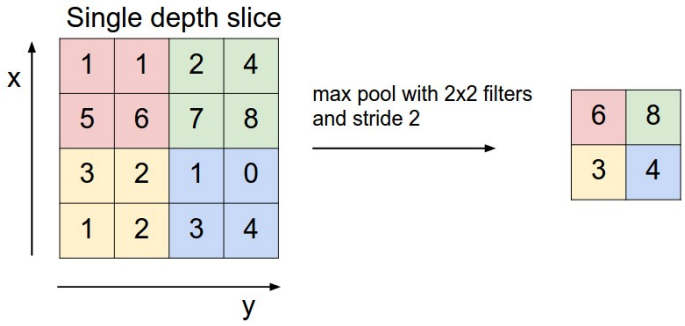
Sdružovací vrstva (*Pooling*) se stará o převzorkování rozlišení (*downsampling*) předchozí vrstvy. Často se používá filtr o rozměru 2x2 typu max-pooling. Vstupní matice se rozdělí na dílčí matice o rozměru 2x2 a z každé z dílčích matic je vybrána pouze maximální hodnota jako hodnota do matice výstupní. Operace je znázorněna na obrázku 3.4. Tím dochází ke snížení rozměru vrstev na polovinu (při zachování počtu kanálů). Praktickým důsledkem užití této vrstvy je snížení počtu spojení mezi neurony vrstev a tím rychlejší zpracování

[3] https://cambridgespark.com/content/tutorials/convolutional-neural-networks-with-keras/



Obrázek 3.3: Znázornění operace konvoluce pomocí filtru K o rozměru 3x3 vstupního obrazu I a výsledek operace $I * K$. Zdroj obrázku[3].

dalších operací. Na rozdíl od konvoluční vrstvy, sdružovací vrstva nemá žádný parametr, který by byl ovlivněn procesem učení [12].



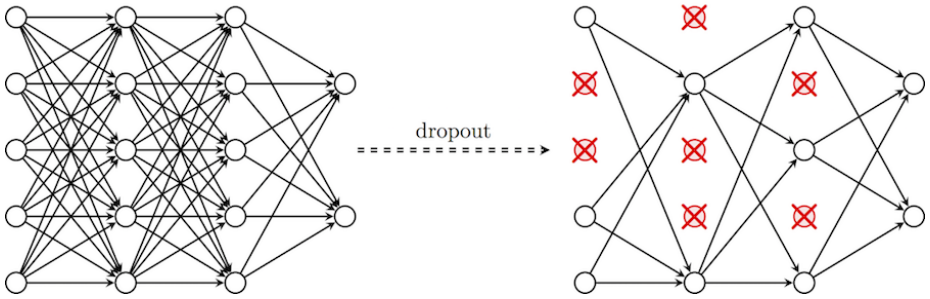
Obrázek 3.4: Operace max pooling s filterem o rozměru 2x2. Vlevo vstupní matice, vpravo výsledek operace. Zdroj obrázku [12].

Plně propojená

Pojem plně propojená vrstva vychází z toho, že každý neuron vrstvy předchozí je propojen s každým neuronem vrstvy následující. Plně propojená vrstva bývá typicky poslední vrstvou konvoluční neuronové sítě a účelem této vrstvy se stává přiřazení jednotlivých příznaků získaných předchozími vrstvami k třídám, které reprezentují. Výstup plně propojené vrstvy se obvykle pojí s aktivační funkcí softmax, která zajišťuje, že součet pravděpodobností všech klasifikovaných tříd je vždy roven jedné. Ukázka sítě s plně propojenými vrstvami je znázorněna na obrázku 3.5 vlevo.

Dropout

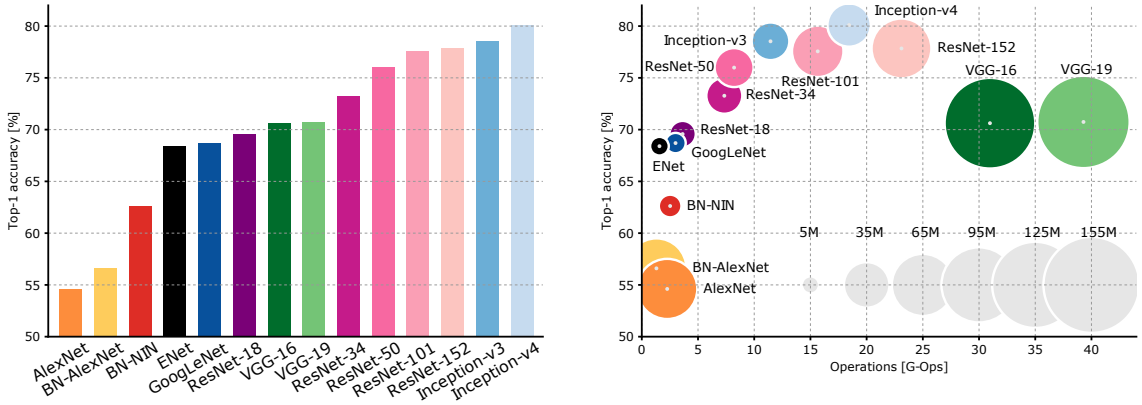
Vrstva dropout se používá v architektuře konvolučních neuronových sítí v kombinaci s jinými vrstvami, především plně propojenou, ale i sdružovací, kde poté tvoří určitý šum. Účelem dropout vrstvy v průběhu každé iterace učení na základě parametru pravděpodobnosti p náhodně deaktivovat některé neurony. Tyto deaktivované neurony se poté neberou v potaz při aplikací konvolučních či sdružovacích filtrů. Praktickým důsledkem aplikace vrstvy dropout je snížení náchylnosti vůči přetrénování sítě a přispívá k lepší generalizaci získaných příznaků [23] Princip operace dropout je znázorněn na obrázku 3.5.



Obrázek 3.5: Znázornění neuronové sítě s plně propojenými vrstvami před a po aplikaci operace dropout na všech vrstvách. Zdroj obrázku[5].

3.2.3 Architektury konvolučních neuronových sítí

Některé důležité architektury konvolučních neuronových sítí dostaly svůj název. Architekturou se v tomto případě rozumí konkrétní uspořádání jednotlivých vrstev a jejich parametrů popsaných v kapitole 3.2.2. Jednotlivé architektury se tak liší počtem vrstev, jejich zastoupením, uspořádáním a parametry a dosahují rozdílné úspěšnosti a výkonnosti v klasifikačních úlohách. Použití již vytvořené architektury pro klasifikaci má výhody v jednoduchosti použití, ověřeném chování sítě a úspěšnosti. První úspěšnou aplikací konvolučních sítí byla síť **LeNet** v 90. letech, která byla použita pro klasifikaci ručně psaných číslic [12]. Na obrázku 3.6 jsou srovnány úspěšnosti některých důležitých architektur konvolučních sítí včetně závislosti úspěšnosti na složitosti sítě (počtu operací při průchodu sítí). Některé z významných architektur sítí, které jsou v této práci použity, jsou popsány níže.

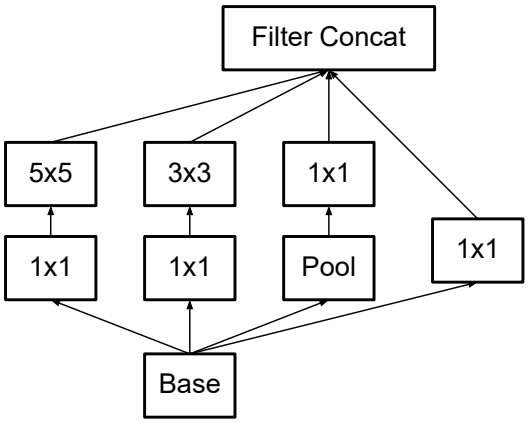


Obrázek 3.6: Srovnání úspěšnosti některých důležitých používaných architektur konvolučních neuronových sítí. **Vlevo**: srovnání úspěšnosti tzv. Top-1 accuracy (výstup s nejvyšší pravděpodobností odpovídá požadovanému výstupu). **Vpravo**: úspěšnost v závislosti na počtu operací potřebných k vykonání jednoho průchodu sítí. Střed kruhů znázorňuje úspěšnost, jeho velikost počet operací. Zdroj obrázku [2].

[5] https://cambridgespark.com/content/tutorials/convolutional-neural-networks-with-keras/

Inception

Základem architektury Inception je její stavební blok modul Inception, jehož základní verze je znázorněna na obrázku 3.7. Jedná se o paralelní kombinaci 1x1, 3x3 a 5x5 konvolučních filtrů, vždy s předzpracováním pomocí 1x1 konvoluce pro redukci množství příznaků. Výhodou modulu Inception je, že není třeba se rozhodovat mezi použitými rozměry konvolučních filtrů v jednotlivých vrstvách a místo toho necháme rozhodnout síť, které příznaky využije [22].

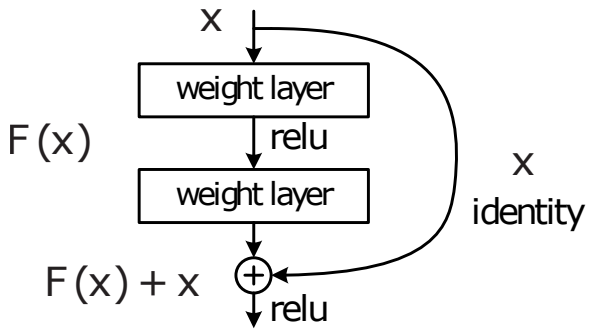


Obrázek 3.7: Základní modul Inception. Zdroj obrázku [22].

ResNet

S rostoucí hloubkou neuronové sítě začínají být problémy s jejím trénováním a přidáváním dalších vrstev se zvyšuje chyba, která není způsobena přetrénováním, ale složitou optimalizací hluboké sítě. Sítě ResNet, z anglického *residual network*, tento problém řeší zavedením reziduálních bloků, které tvoří základní stavební blok těchto sítí (viz obrázek 3.8).

Nechť x je vstup podsítě a $\mathcal{H}(x)$ je její skutečný výstup. Zbytková funkce je rozdíl mezi nimi $\mathcal{F}(x) = \mathcal{H}(x) - x$, původní funkci je tak možné vyjádřit ve tvaru $\mathcal{H}(x) = \mathcal{F}(x) + x$. Přesto, že jsou oba zápisy zaměnitelné, složitost optimalizace při trénování zbytkové funkce je odlišná [10]. Zbytkovou funkci $\mathcal{F}(x) + x$ je možné realizovat pomocí dopředného spojení znázorněného na obrázku 3.8.



Obrázek 3.8: Stavební blok sítě ResNet – reziduální blok. Zdroj obrázku [10].

3.3 Klasifikace třídy objektu

Nejvíce přímočaré využití konvolučních neuronových sítí je pro klasifikaci typu objektu. Na vstup konvoluční sítě je přiveden vstupní obraz v podobě matice reprezentující hodnoty pixelů pro každý barevný kanál s předem definovaným rozlišením. U vstupního obrazu již implicitně předpokládáme, že zobrazuje právě jednu z klasifikovaných tříd a v ideálním případě co nejméně rušivých a matoucích elementů jako jsou přebytečné okraje, další nesouvisející objekty a podobně.

Výstup sítě je tvořen N výstupními neurony odpovídající N klasifikovaným třídám objektů. Výstup je typicky tvořen plně propojenou vrstvou se softmax aktivační funkcí, která zajišťuje, že součet pravděpodobností všech výstupů bude roven 1. Po zavedení vstupu je proveden průchod sítí a na výstupních neuronech se projeví pravděpodobnosti příslušnosti objektu k jednotlivým klasifikovaným třídám. Výstup s největší pravděpodobností je považován za výslednou třídu klasifikace.

3.4 Detekce objektu

Pro účely detekce objektů v obraze se používají speciální konvoluční neuronové sítě. V práci *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks* [18] byla představena vylepšená verze původní metody *R-CNN – Rich feature hierarchies for accurate object detection and semantic segmentation* [8].

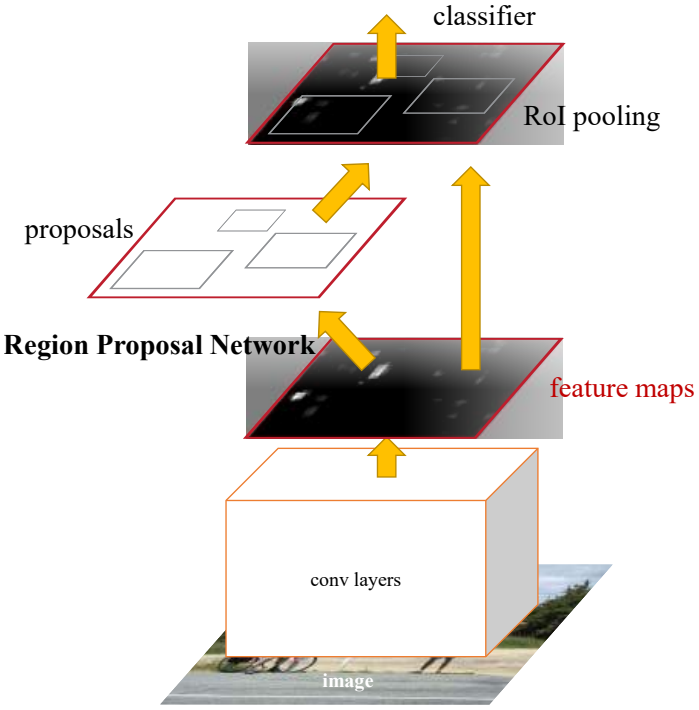
Jádrem fungování sítě je *RPN – Region Proposal Network*. Vstupem této sítě je poslední výstupní konvoluční příznaková mapa obrazu libovolného rozlišení a výstupem návrhy regionů detekovaných objektů včetně odpovídajících pravděpodobností. Blokové schéma architektury sítě je znázorněno na obrázku 3.9. Proces získání regionů probíhá tak, že jsou definovány kotvy – *Anchors* – předem daný počet fixních regionů rovnoměrně rozložených po celém obraze. Ty mají různé velikosti a poměry stran, ve výchozí konfiguraci 3 různé velikosti a 3 různé poměry stran, to odpovídá výsledným 9 kotvám pro každou pozici.

Všechny kotvy jsou následně klasifikovány pomocí dvou různých plně propojených vrstev pro získání dvou typů informací. Prvním je pravděpodobnost přítomnosti objektu pod regionem kotvy bez ohledu na jeho konkrétní třídu (*objectness score*). Druhým regrese pro úpravu parametrů kotvy tak, aby její region odpovídal případnému detekovanému objektu. Aplikací regrese dostáváme finální návrhy detekovaných regionů, které následně prochází procesem klasifikace pro získání konkrétní třídy objektu.

3.5 Detekce kontury objektu

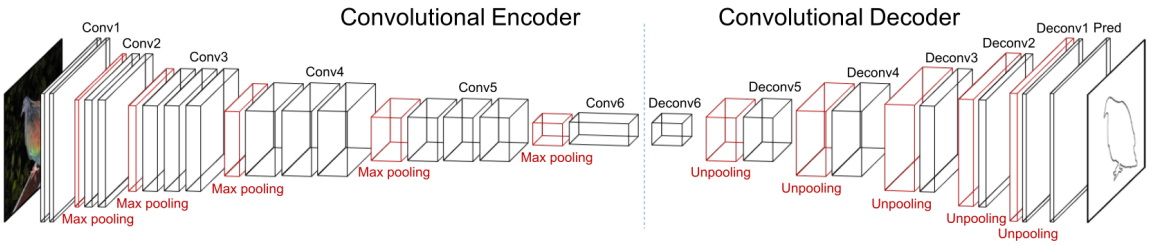
Práce *Object Contour Detection with a Fully Convolutional Encoder-Decoder Network* [24] popisuje využití hluboké plně konvoluční enkodér dekodér neuronové sítě pro detekci hran objektu se zaměřením na hrany ohraničující celý objekt vůči pozadí.

První částí sítě je konvoluční enkodér skládající se ze 6 konvolučních vrstev inspirovaných prvními vrstvami architektury sítě VGG-16. Následuje část sítě referovaná jako konvoluční dekodér, kde první vrstva slouží k redukci dimenze filtrů a posléze 5 unpooling dekonvolučních vrstev, které se starají o opětovné zvýšení rozlišení obrazu. Schéma sítě je znázorněno na obrázku 3.10. Výsledkem je maska kontury objektu poloviničního rozlišení původního obrazu, která je tvořena hodnotami v intervalu $\langle 0, 1 \rangle$, kde 0 značí absenci kontury a 1 její přítomnost.



Obrázek 3.9: Blokové schéma sítě Faster-RCNN pro detekci objektů v obraze. Zdroj obrázku [18].

Síť se ukázala být vhodná pro použití obecného charakteru, dosahuje dobrých výsledků na libovolných datech i mimo trénovací množinu. Díky tomu lze použít již natrénovanou síť bez dalších úprav pro detekci kontury na vlastních datech.



Obrázek 3.10: Schéma architektury sítě *fully convolutional encoder-decoder network* pro detekci hran v obraze. Zdroj obrázku [24].

Kapitola 4

Návrh a implementace systému klasifikace vozidel

V této kapitole je popsán návrh a implementace systému klasifikace typu vozidel. Základní verze systému se skládá z detekce vozidla v obraze z dohledové kamery a následné využití detekovaného regionu pro klasifikaci jejich typu. Tato základní verze slouží k ověření koncepce a navazuje na ní rozšířená verze inspirovaná normalizační metodou *Unpack* z práce *BoxCars* [21]. Pro aplikaci metody *Unpack* je potřeba mít k dispozici rozšířené informace o pozorovaném vozidlu, především jeho konturu a odhad směru k úběžníkům scény. Z těchto informací je sestaven odhad 3D bounding boxu vozidla jehož rozložením do rovinné podoby získáme požadovaný normalizovaný obraz.

4.1 Detekce vozidel

Pro účely detekce vozidel je použito *Tensorflow Object Detection API*[1], které poskytuje rozhraní pro trénování vlastních konvolučních detektorů objektů s využitím *Faster R-CNN* sítí blíže popsaných v kapitole 3.4.

Pro vlastní trénování slouží skript `train.py`, který je součástí knihovny. Skript očekává vstupní data pro trénování a vyhodnocení detektoru ve formátu `TF record` a konfigurační soubor s parametry pro trénování `pipeline.config`.

4.1.1 Dataset COD2OK

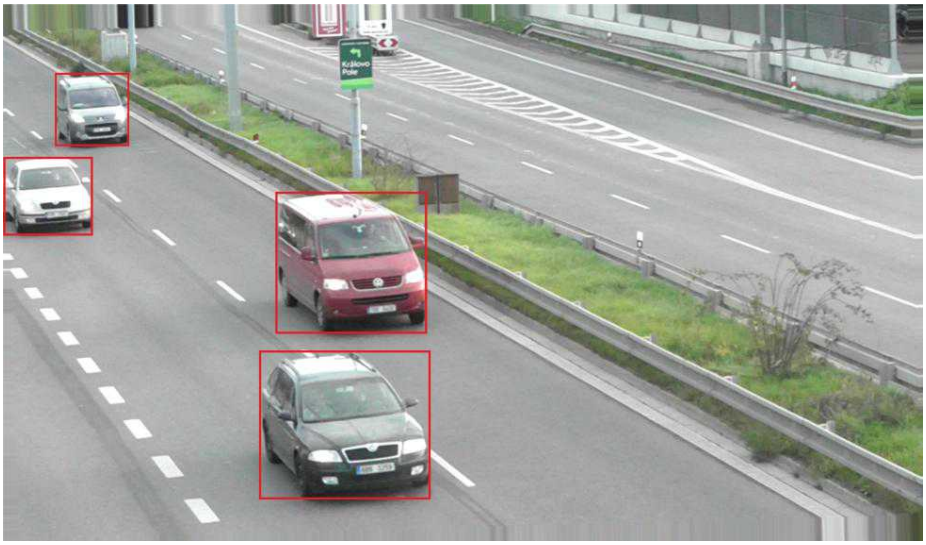
Pro účel detekce byl zvolen dataset COD20K [19], který obsahuje 10 000 trénovacích snímků s 20 000 anotovanými vozidly a dalších 1 100 obrázků s 3 000 instancí vozidel pro vyhodnocení. Dataset je vhodný pro účely detekce, jelikož byl k tomuto účelu navrhnut a obsahuje záběry z různých pozorovacích úhlů odpovídající běžným dohledovým kamerám.

Jednotlivá vozidla jsou anotována pomocí záznamů ve formátu `bbGt`. Jedná se o prostý anotační formát, který obsahuje především třídu a region anotovaného objektu. Vozidla datasetu jsou rozdělená do 4 samostatných tříd:

- `car` – vozidlo jedoucí směrem od kamery (viditelná zadní část).
- `car_from` – vozidlo jedoucí směrem ke kameře (viditelná přední část).
- `rail` – jako `car`, ale část vozidla překrývají svodidla vozovky.
- `rail_from` – jako `car_from`, ale část vozidla překrývají svodidla vozovky.

[1] https://github.com/tensorflow/models/tree/master/research/object_detection

Třídy vozidel jsou pro účely trénování zachovány kvůli potenciálně lepším výsledkům díky menší variabilitě uvnitř tříd, dále se s nimi však již nepracuje a v potaz se berou pouze detekované regiony. Na obrázku 4.1 je znázorněna Ukázka snímku z datasetu COD20K.



Obrázek 4.1: Ukázka snímku z datasetu COD20K včetně znázorněných anotací v podobě regionů výskytu vozidel.

4.1.2 Trénování detektoru

Proces trénování detektoru se skládá z přípravy trénovacích dat, konfigurace parametrů a vlastního trénování.

Pro účel přípravy dat vznikl skript `create_cod20k_tf_record.py` jehož vstupem je dataset COD20K a výstupem dvojice souborů `train_cod20k.record` a `eval_cod20k.record`. Tyto soubory formátu *TensorFlow record* obsahují serializovaná obrazová data a anotace vozidel ve formátu, který očekává trénovací skript detektoru.

Soubor `pipeline.config` slouží ke konfiguraci a úpravě parametrů trénování detektoru. Z důvodu komplexnosti se zde nenachází kompletní přehled, ale pouze některé důležité parametry:

- `num_classes` – kolik různých tříd objektů se bude detekovat (počet výstupů sítě).
- `image_resizer` – rozlišení, na které bude obraz převzorkován během trénování.
- `feature_extractor` – architektura sítě pro extrakci příznaků obrazu.
- `grid_anchor_generator` – parametry generování kotev detektoru.
- `batch_size` – počet snímků paralelně zpracovaných v jedné iteraci.
- `learning_rate` – parametr rychlosti učení.
- `fine_tune_checkpoint` – výchozí váhy trénovaného modelu.
- `num_steps` – počet iterací trénovacího algoritmu.
- `train_input_reader` – soubor obsahující data pro trénování.
- `eval_input_reader` – soubor obsahující data pro vyhodnocení.

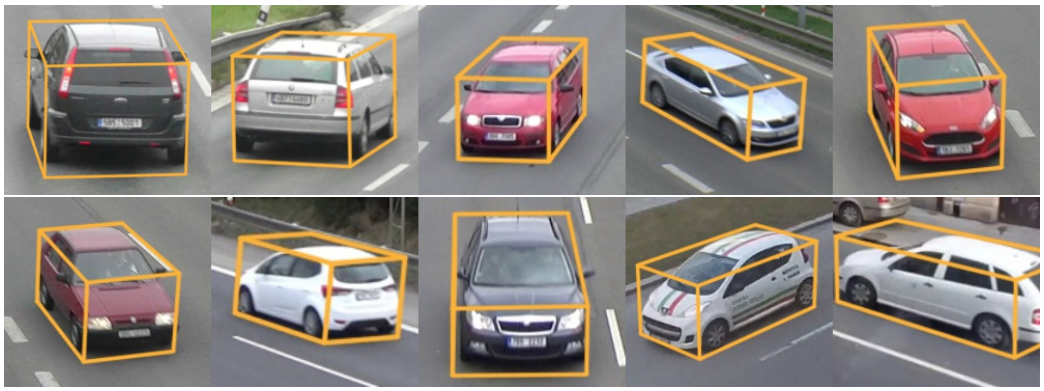
4.2 Klasifikace třídy vozidla

Pro proces klasifikace jsou využity dva různé přístupy, které jsou následně srovnány podle jejich úspěšnosti. Prvním je přímá klasifikace, kdy je výstup detektoru vozidel bez dodatečných úprav přímo využit jako vstup konvoluční neuronové sítě pro zjištění typu vozidla. Druhý přístup je inspirován metodou *Unpack* z práce *BoxCars* [21]. Účelem metody *Unpack* je normalizace vstupních dat klasifikátoru a její přesný princip je popsán níže.

Implementace je realizována pomocí knihovny *Keras: The Python Deep Learning library*[2], která poskytuje API vysoké úrovně abstrakce nad knihovnou strojového učení *Tensorflow*[3]. Cílem knihovny *Keras* je poskytnutí rozhraní pro rychlé prototypování systémů konvolučních neuronových sítí.

4.2.1 Dataset BoxCars116k

Dataset z práce BoxCars [21] obsahující 116 000 instancí vozidel 693 různých tříd zachycených ze 137 různých kamer s velkou variabilitou pozorovacích úhlů (viz obrázek 4.2). Kromě informace o třídě vozidla je součástí datasetu i 2D a 3D bounding box vozidla. Díky těmto vlastnostem se jedná o vhodný dataset pro účely klasifikace typu vozidel s využitím navrhované přímé i normalizované metody.



Obrázek 4.2: Ukázka vozidel z datasetu BoxCars116k pořízených z různých úhlů pohledu se znázorněním 3D bounding boxů.

4.2.2 Data augumentation

Předzpracování obrazu při procesu trénování konvoluční sítě hraje potenciální roli ve výsledné úspěšnosti trénovaného modelu a předchází přetrénování sítě. Principem je při každém průchodu trénování náhodná úprava použitého obrazu a tím syntetické rozšíření datasetu. V práci jsou použity dvě metody, které jsou vybrány s ohledem na aplikaci v oblasti klasifikace typu vozidel.

[2] https://keras.io/
[3] https://www.tensorflow.org/

Horizontální převrácení

Operace `flip` vytváří symetrické pozorovací podmínky pro záběry z vnitřní a vnější strany vozovky a eliminuje tak případné nevyváženosti v trénovacích datech (viz obrázek 4.3). Při trénování je operace `flip` aplikována s pravděpodobností 50%.

Obrázek 4.3: Ukázka horizontálního převrácení obrazu pro účely syntetického rozšíření datasetu při trénování.

Změna barvy

Změna barvy pomocí operace `color_change` poskytuje robustnost klasifikátoru vůči konkrétní barvě vozidla. Změna barvy probíhá převodem standardního *RGB* obrazu do barevného modelu *HSV*, ten totiž poskytuje jednodušší manipulaci s barevným spektrem. Následně je aplikována náhodná aditivní změna jeho jednotlivých složek pro každý pixel (viz obrázek 4.4). Barevný model *HSV* reprezentuje:

- **Hue** – odstín barvy.
- **Saturation** – sytost (množství bílého světla).
- **Value** – hodnota jasu (světlost).

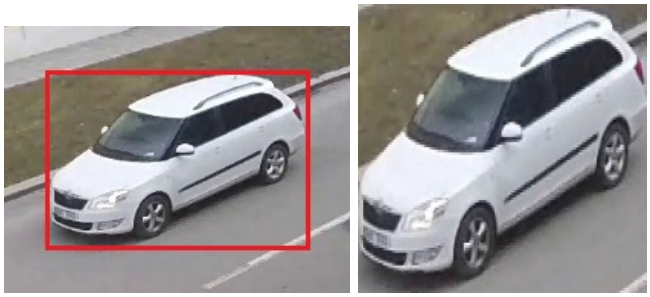
Obrázek 4.4: Ukázka předzpracování obrazu pomocí operace změny barvy. Vlevo originální obraz, dále různé náhodné výsledky operace.

4.2.3 Přímá klasifikace

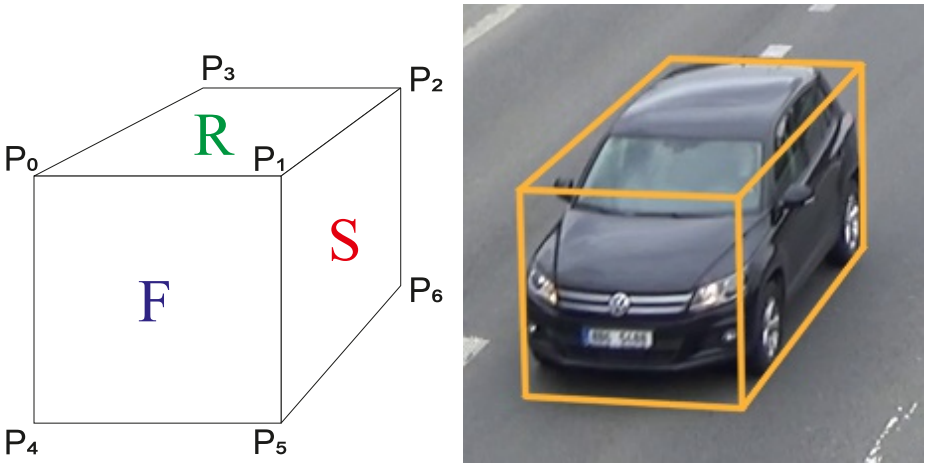
Do procesu přímé klasifikace vstupují snímky vozidel ve své standardní podobě. Při trénování se jedná o region určený anotací datasetu, při testování region detekce vozidla. V obou případech je daný region transformován na očekávané rozlišení a následně použit jako vstup konvoluční sítě, jak je znázorněno na obrázku 4.5. V případě trénování je vstup předzpracován pomocí operací `flip` a `color_change`.

4.2.4 Klasifikace s využitím Unpack

Do procesu klasifikace s normalizační metodou *Unpack* vstupuje 3D bounding box vozidla. Pomocí obrazových transformací je obraz normalizován do rovinné podoby a tím jsou mini-



Obrázek 4.5: Ukázka metody přímé klasifikace vozidel. **Vlevo:** region detekce vozidla, **vpravo:** region upravený na vstup konvoluční sítě.



Obrázek 4.6: Znázornění 3D bounding boxu vozidla. **Vlevo:** reprezentace 3D bounding boxu pomocí sedmi bodů souřadnic rohů. **Vpravo:** 3D bounding box konkrétního vozidla z datasetu BoxCars116k.

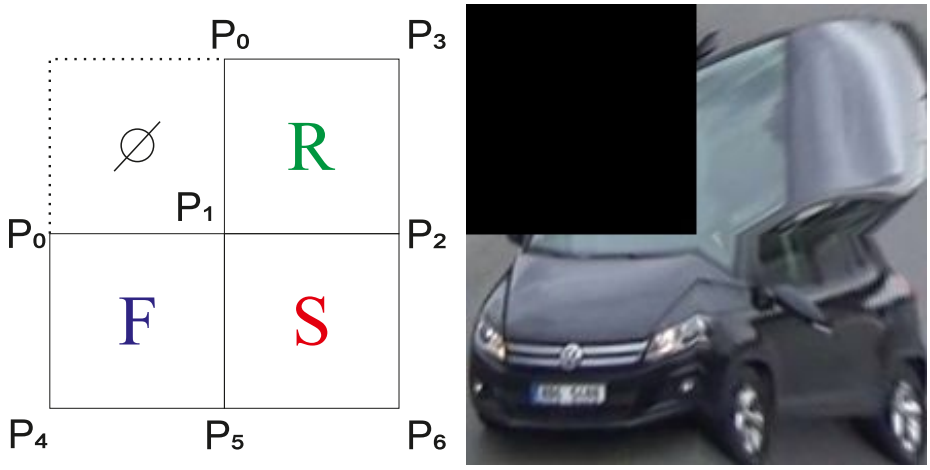
malizovány rozdílné pozorovací úhly scény. Tento postup je použit jak pro proces trénování klasifikátoru, tak i při testovací fázi klasifikace. Metoda *Unpack* je doplněna o předzpracování dat operací `color_change`.

3D bounding box

Reprezentuje původní 3D scénu, která byla při pořízení snímku převedena do 2D reprezentace. 3D bounding box je reprezentován jako sedm uspořádaných bodů $P_i = (x_i, y_i)$, kde každý bod vyjadřuje souřadnice konkrétního rohu bounding boxu (viz obrázek 4.6). Pro účely trénování jsou informace o 3D bounding boxu vozidla součástí datasetu *BoxCars116k*, v testovací fázi klasifikace tyto informace nejsou k dipozici a budou odhadnuty z obrazových dat. Postup sestavení odhadu 3D bounding boxu vozidla z informací o kontuře a směru k úběžníkům je popsán v kapitole 4.5.

Unpack

Metoda pracuje s reprezentací 3D bounding boxu a jejím cílem je nalezení transformace zobrazení perspektivy původního obrazu 3D bounding boxu na jeho zdánlivou rovinnou reprezentaci. Výsledný obraz je tvořen čtyřmi částmi, které reprezentují příď/záď, střechu



Obrázek 4.7: Znázornění transformace 3D bounding boxu pomocí metody Unpack. **Vlevo:** Reprezentace 3D bounding boxu a jeho bodů po aplikaci metody Unpack. **Vpravo:** Unpack 3D bounding boxu konkrétního vozidla z datasetu BoxCars116k.

a bok vozidla a nulový segment pro doplnění na obdélník (viz obrázek 4.7). Díky tomu dochází k minimalizaci rozdílů mezi snímky z různých pozorovacích úhlů a jednotlivé klíčové části vozidel se po aplikaci metody *Unpack* nachází vždy ve stejné části obrazu. Operace je realizována pomocí funkce `findHomography` knihovny `OpenCV`[4].

4.3 Odhad směru k úběžníkům vozidla

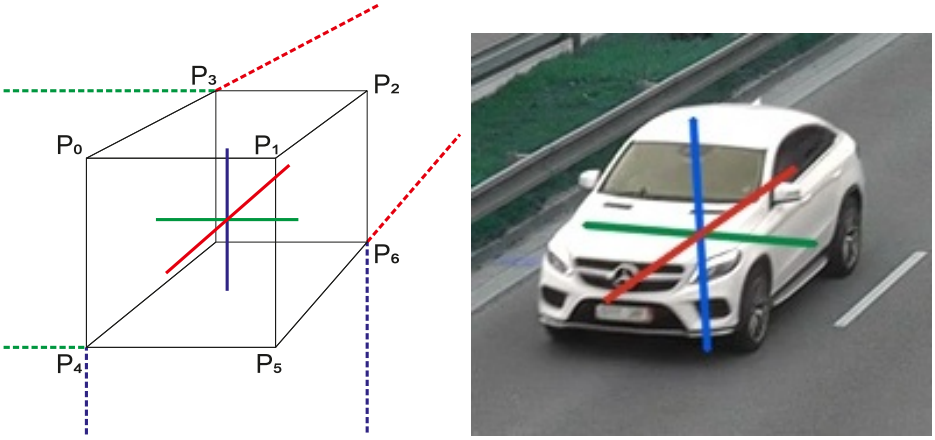
Odhad směru k úběžníkům od vozidla je realizován jako klasifikační úloha pomocí samostatné konvoluční neuronové sítě s využitím datasetu *BoxCars116k*. Principem je při trénovací fázi převod 3D bounding boxu vozidla na informaci o směru k úběžníku konkrétní dimenze obrazu od středu vozidla. Výsledkem jsou 3 samostatné úhly v intervalu $\langle -90°, 90° \rangle$, viz obrázek 4.8 vlevo, které určují odchylku směru k úběžníku od svislé nebo vodorovné osy obrazu. Úhly jsou převedeny na diskrétní binární hodnoty s rozestupem 3°. Tyto binární hodnoty lze poté využít jako výstupy konvoluční neuronové sítě. Výsledná konvoluční neuronová síť má tedy 3 paralelní výstupní vrstvy odpovídající 3 dimenzím obrazu, každá po 60 výstupech odpovídající úhlům $(-90°, -87°, -84°, \ldots, 87°, 90°)$ a určující pravděpodobnost úhlu směru k úběžníku.

V testovací (provozní) fázi je pak výstup detektoru vozidel v podobě regionu detekce vozidla podroben klasifikaci směru k úběžníkům a nejpravděpodobnější výstup každé dimenze je převeden na svou úhlovou reprezentaci pro potřeby sestavení 3D bounding boxu vozidla. Výsledek procesu je znázorněn na obrázku 4.8 vpravo.

4.4 Detekce kontury vozidla

Pro potřeby detekce kontury vozidla je využito konvoluční sítě z práce *Object Contour Detection with a Fully Convolutional Encoder-Decoder Network* [24] blíže specifikované v kapitole 3.5. Vstupem sítě je bounding box detekce rozšířený v každém směru o 10% svých

[4] https://opencv.org/

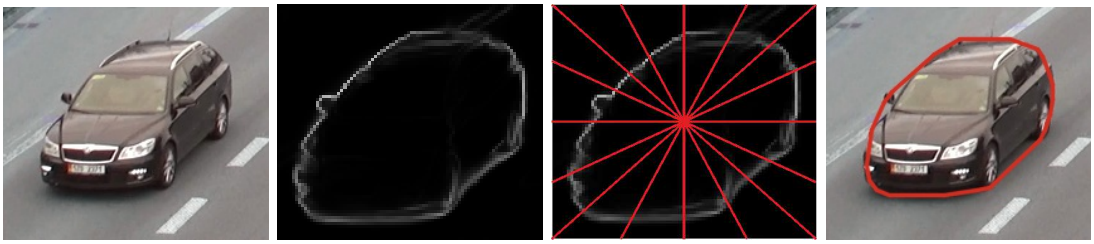


Obrázek 4.8: Znázornění odhadu směru k úběžníkům od vozidla. **Vlevo:** proces získání směru k úběžníkům z 3D bounding boxu vozidla pro trénování klasifikátoru. **Vpravo:** rasterizované směry k úběžníkům konkrétního vozidla.

původních rozměrů. Důvodem rozšíření je zlepšení detekce hran, které se nachází v těsné blízkosti hranice obrazu.

Výstupem konvoluční sítě je maska poloviního rozlišení původního obrazu, s hodnotami pixelů v intervalu $\langle 0, 1 \rangle$, vyjadřující pravděpodobnost přítomnosti hrany v konkrétním bodě. Postup získání samotné kontury z masky, který je znázorněn na obrázku 4.9, probíhá následovně:

- Nechť S je střed regionu detekce vozidla a B_i body ležící na hraně detekovaného regionu tak, že v každém z rohů leží jeden bod a na každé hraně orazu N bodů, které mezi sebou mají rovnoměrné rozestupy.
- Pak získané úsečky mezi středem a každým z hraničních bodů – SB_i – mají tu vlastnost, že vždy jeden konec leží uvnitř a druhý konec vně vozidla.
- Pro každou z úseček je nalezen bod C_i, jako bod na úsečce, ve kterém se nachází maximální hodnota masky.
- Pro body konturu tvořené body C_i je nalezen její konvexní obal a tento použit jako výsledná kontura vozidla pro účely sestavení 3D bounding boxu.



Obrázek 4.9: Proces získání kontury vozidla. **Zleva postupně:** region detekce vozidla, výstupní maska detektoru kontury, úsečky pro hledání bodů kontury (maximální hodnoty masky), výsledná konvexní kontura vozidla.

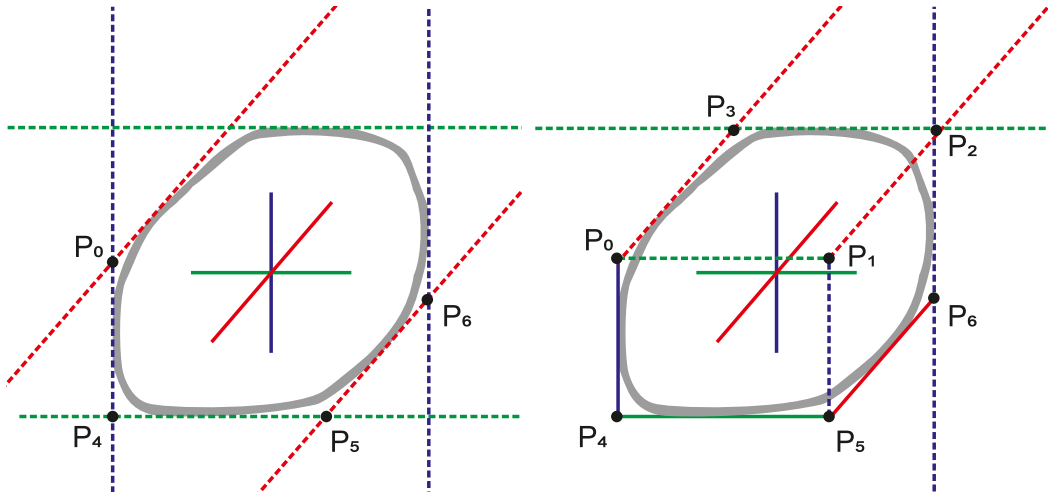
4.5 Konstrukce 3D bounding boxu

Principem sestavení 3D bounding boxu vozidla je aplikace získaných informací o směru k úběžníkům a konvexní kontuře vozidla. Postup konstrukce 3D bounding boxu, který je znázorněn na obrázku 4.10 je následující:

- Pro každý směr k úběžníku jsou nalezeny obě tečny s konvexní konturou vozidla rovnoběžné se směrem k úběžníku.
- Na základě průsečíků odpovídajících tečen jsou pak definovány body P_0, P_4, P_5 a P_6 3D bounding boxu (viz obrázek 4.10 vlevo).
- Bod P_1 je získán pomocí pomocných tečen z bodu P_0 a P_6
- Pomocí pomocné tečny z bodu P_1 získáváme bod P_2
- Pomocí pomocné tečny z bodu P_2 získáváme bod P_3
- Dostáváme finální 3D bounding box (viz obrázek 4.10 vpravo).

Z obrázku 4.10 může vyplývat,že body P_2 a P_3 by mohly být získány pomocí průsečíků původních teček kontury, protože pomocné tečny jsou shodné s původními. V testovacím provozu se však projevovaly chyby odhadu kontury, vůči kterým je popsaný postup robustnější.

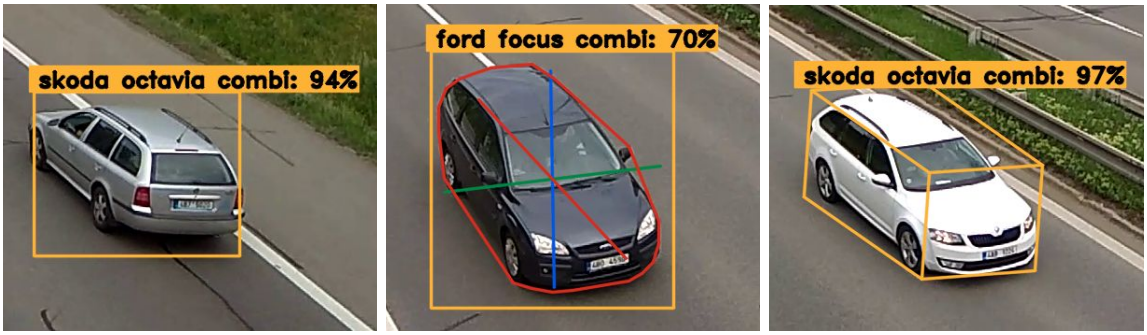
Konstrukce 3D bounding boxu nebere v potaz reálnou perspektivu obrazu a pracuje se směrem k úběžníku, který má zdánlivě nekonečnou vzdálenost, přesto že reálná vzdálenost úběžníku je rozdílná. Vzhledem k typické velikosti vozidel vůči scéně se však zásadně neprojevuje nepřesnost konstrukce 3D bounding boxu, kterou zanedbání perspektivy působí.



Obrázek 4.10: Znázornění konstrukce 3D bounding boxu vozidla. **Vlevo:** konvexní kontura vozidla s tečnami rovnoběžnými se směry k úběžníkům a body 3D bounding boxu, které lze získat jejich průsečíky. **Vpravo:** kompletní 3D bounding box získaný pomocnými tečnami vedenými ze získaných bodů.

4.6 Výsledná aplikace

Výsledná aplikace spojuje všechny výše popsané postupy do kompletního systému klasifikace typu vozidel. Vstupem aplikace jsou jednotlivé snímky nebo video z dohledové kamery všech formátů podporovaných knihovnou *OpenCV* (`.bmp`, `.jpg`, `.png`, `.bmp`, `.mp4`, `.avi`, ...). Výstupem aplikace je pak stejná množina snímků nebo videí, doplněných o anotace získaných informací, které lze konfiguračně modifikovat. Na obrázku 4.11 je znázorněno několik různých výstupů aplikace s rozdílnou konfigurací.



Obrázek 4.11: Ukázka výstupů aplikace znázorňující klasifikovanou třídu vozidla a pravděpodobnost určení s různými konfiguracemi anotací obrazu. **Vlevo:** region detekce, **uprostřed:** včetně kontury a směru k úběžníkům, **vpravo:** odhad 3D bounding boxu vozidla.

4.6.1 Architektura aplikace

Aplikace se skládá ze 6 samostatných vláken, které jsou vzájemně propojeny frontami, pomocí kterých si předávají zpracovaná data reprezentovaná instancí třídy `Passage`. Každé vlákno využívá informace získané v předchozích vláknech a postupuje v procesu zpracování. Kompletní schéma architektury aplikace je znázorněno na obrázku 4.12. V následujícím přehledu jsou sepsány jednotlivá vlákna zpracování a jejich zodpovědnost:

- **Thread Input** – čtení a dekódování komprimovaných obrazových dat (snímky/video). Výstupem jsou obrazové matice pixelů.
- **Thread Detection** – detekce vozidel. Výstupem je množina detekovaných regionů.
- **Thread Angles Estimation** – odhad směru k úběžníkům. Výstupem je trojice úhlů odpovídající směrům k úběžníkům pro každé detekované vozidlo.
- **Thread BB3D Estimation** – odhad 3D bounding boxů vozidel, probíhá detekce kontury vozidla. Výstupem je zkonstruovaný 3D bounding box pro každou detekci.
- **Thread Classification** – klasifikace typu vozidel pomocí aplikace metody Unpack. Výstupem jsou třídy vozidel (ID třídy a textová informace názvu třídy) a pravděpodobnost náležitosti třídě.
- **Thread Output** – výstupní vlákno, které zajišťuje finální anotace obrazu a zpětné ukládání snímků nebo videí na disk.

Součástí aplikace jsou další dvě pomocné třídy:

- **Passage** – Třída reprezentující jeden snímek zpracování. Obsahuje obrazová data a veškeré informace zjištěné během zpracování. Instance třídy `Passage` prochází všemi vlákny zpracování skrze fronty pro předávání dat.
- **TFModelBase** – třída, ze které dědí ostatní třídy využívající klasifikační modely. zajišťuje opakující se operace načtení modelu a správu sezení.

Konfigurace

Aplikace disponuje jednoduchým uživatelským konfiguračním aparátem `config.py`, pomocí kterého lze upravovat parametry klasifikace a konfigurovat výstupy aplikace. Popis jednotlivých položek je součástí přílohy A.

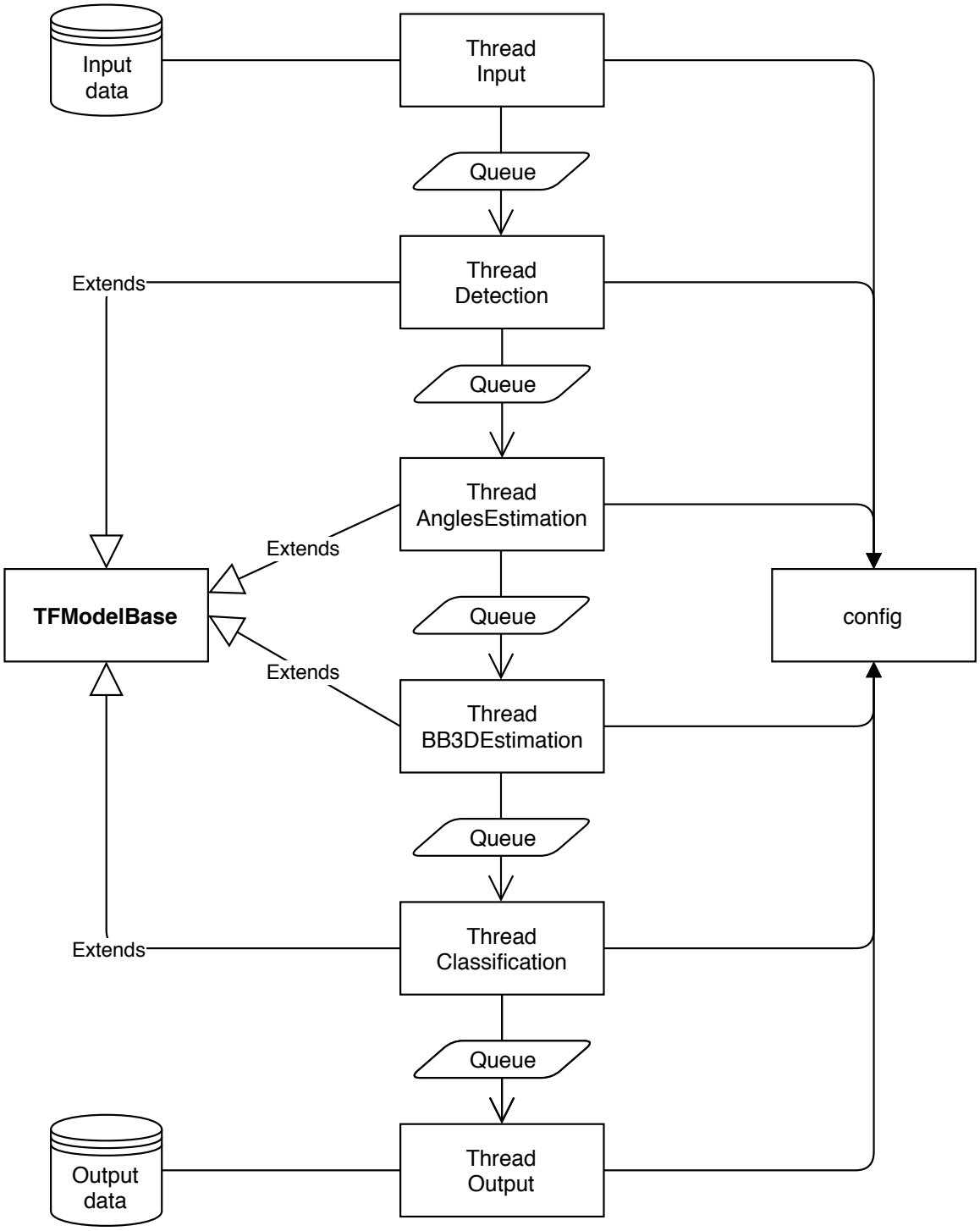
4.6.2 Možná rozšíření aplikace

Během tvorby a testování aplikace se vyskytly skutečnosti, které nebyly při návrhu aplikace známé a nemají vliv na samotnou funkčnost. Tato rozšíření jsou ponechána jako možná pokračování implementace tohoto systému.

Do procesu detekce vozidel v tuto chvíli vstupuje vždy celý obraz. Často však část obrazu nemá smysl podrobovat zpracování, protože se jedná o příliš vzdálené objekty nebo přilehlé okolí vozovky. Definicí regionu zájmu pro detekci vozidel by přispělo k rychlejšímu zpracování detekce vozidel snímku a ušetření dalšího zpracování detekcí vozidel, které jsou v záběru příliš vzdálené a nejsou dostatečně viditelné pro úspěšnou klasifikaci jejich typu.

Dalším možným rozšířením by bylo zařazení funkce sledování objektu (*tracking*) při zpracování videa. Současná aplikace funguje čistě jako jednosnímková a nepřenáší žádné informace mezi jednotlivými snímky. To má za důsledek, že ve výsledném videu se často mezi jednotlivými snímky mění klasifikovaný typ vozidla, což nepůsobí příliš uživatelsky přívětivě. Možným řešením by bylo sledování jednotlivých vozidel na jejich trajektorii a zobrazovat pouze nejlepší výsledek klasifikace z celé série snímků.

Posledním zmíněným potenciálním rozšířením aplikace by byla funkce zpracování živého streamu z dohledové kamery. Současná aplikace pracuje pouze se záznamem, to je výhodné pro vývoj a testování, ale pro reálný provoz dohledové aplikace by se očekávalo přímé napojení na obraz kamery a zpracování dat v reálném čase. Toho by šlo docílit využitím dostatečně výkonného hardwaru s podporou grafické akcelerace *CUDA* a napojením na stream kamery pomocí knihovny *OpenCV*.



Obrázek 4.12: Blokové schéma aplikace znázorňující jednotlivá vlákna zpracování, synchronizační fronty pro předávání dat, vstupy a výstupy.

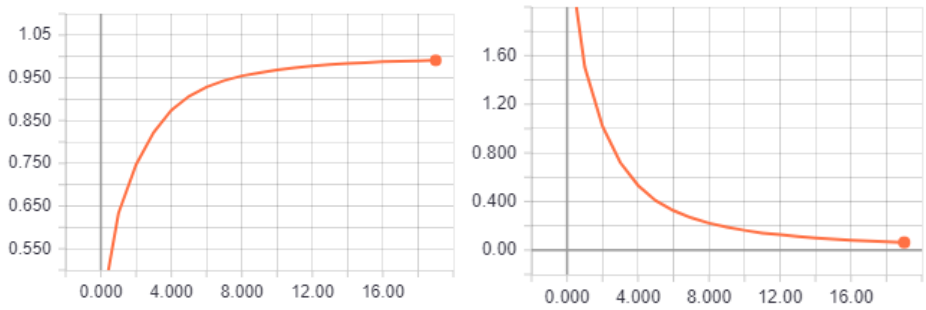
Kapitola 5

Vyhodnocení úspěšnosti systému

Následující kapitola se zaměřuje na vyhodnocení úspěšnosti jednotlivých částí systému klasifikace typu vozidel. Vyhodnocení bylo z větší části provedeno automatizovaně na testovacích množinách využitých datasetů. Tyto testovací množiny nebyly využity pro účely trénování a poskytnuté výsledky by tak neměly být zkreslené. Jedna část vyhodnocení byla provedena ručně nad videem z dohledové kamery bez přítomnosti anotací, pouhým sledováním průjezdů a výsledků systému.

5.1 Konvergence trénování

Průběh trénování jednotlivých klasifikátorů byl sledován pomocí aplikace *TensorBoard*[1]. Sledováním průběhu funkcí Loss a Accuracy byla vždy ověřena konvergence trénovaného modelu a byl zvolen vhodný bod ukončení procesu učení. Na obrázku 5.1 je znázorněna ukázka průběhu trénování klasifikátoru typu vozidel s využitím metody Unpack.



Obrázek 5.1: Ukázka vizualizace z aplikace TensorBoard, průběhu trénování klasifikátoru typu vozidel. Osa X znázorňuje iteraci trénování (epocha), osa Y hodnotu sledované funkce. **Vlevo:** Accuracy, **vpravo:** Loss.

[1] https://www.tensorflow.org/programmers_guide/summaries_and_tensorboard

5.2 Úspěšnost detekce vozidel

Úspěšnost detekce vozidel je vyhodnocena pomocí standardní metriky *AP – average precision*[2]. Pro započítání pozitivní detekce musí být parametr *IOU – Intersection over union*[3] detekovaného regionu alespoň 50%. Vyhodnocení proběhlo na testovací množině datasetu COD20K, obsahující okolo 1100 snímků s 3000 instancí vozidel. Vyhodnocení je založeno na trénovaném modelu *Faster R-CNN InceptionV2*.

Tabulka 5.1: Úspěšnost detekce vozidel.

Třída detekce	AP – average precision
`car`	0.7064
`car_from`	0.7313

5.3 Úspěšnost odhadu směru k úběžníkům scény

Odhad směru k úběžníkům byl vyhodnocen na testovací množině datasetu BoxCars116k obsahující okolo 40 000 vzorků. Vyhodnocení bylo provedeno nad modely InceptionV3 a ResNet50, ale vzhledem k tomu, že výsledky byly prakticky zaměnitelné, jsou zobrazeny pouze výsledky modelu InceptionV3. Základním vyhodnocením odhadu směru k úběžníkům je Top-1 accuracy specifikované zvlášť pro jednotlivé úběžníky. Kromě toho nás také zajímá jak velké se model dopouští odchylky oproti očekávané hodnotě. Odhad směru k úběžníkům totiž nelze segmentovat na diskrétní Správně/Špatně, ale zajímá nás také jak daleko je odhad od skutečné hodnoty.

Pro toto vyhodnocení byla využita modifikovaná verze Top-1 accuracy „neighbor hit“, která akceptuje jako správný výsledek i sousední hodnoty, které se liší o 3°. Tímto přístupem bylo ověřeno, že značná část chybných výstupů je stále blízká požadované hodnotě a lze je úspěšně použít pro klasifikaci. Jako poslední byla spočítána průměrná odchylka odhadnuté a skutečné hodnoty, která nám může více přiblížit přesnost klasifikátoru. Všechny výsledky jsou zobrazeny v tabulce 5.2.

Tabulka 5.2: Úspěšnost klasifikace odhadu směru k úběžníkům.

Úběžník	Top-1	Top-1 + neighbor	Průměrná odchylka
X	79.60%	88.34%	2.06°
Y	82.64%	93.93%	0.78°
Z	79.24%	96.18%	1.05°

5.4 Úspěšnost klasifikace typu vozidel

Vyhodnocení úspěšnosti obou implementovaných přístupů ke klasifikaci typu vozidel, přímé a s využitím normalizační metody Unpack. Oba porovnané modely využívají sít InceptionV3 trénovanou se stejnými parametry pro co nejvěrohodnější výsledek. Vyhodnocení probíhá pomocí metriky Top-1 accuracy – bere se v potaz pouze nejvíce pravděpodobná třída

[2] http://host.robots.ox.ac.uk/pascal/VOC/voc2007/devkit_doc_07-Jun-2007.pdf
[3] https://en.wikipedia.org/wiki/Jaccard_index

vozidla a ta musí odpovídat skutečné třídě. Vyhodnocení proběhlo na testovací množině datasetu BoxCars116k obsahující okolo 40 000 vzorků. Testovací množina nebyla využita při trénování modelů.

Hodnota Accuracy udává množství úspěšně klasifikovaných vozidel z celkového počtu testovaných. Hodnota Track Accuracy udává kolik vozidel bylo alespoň jednou úspěšně klasifikováno, pokud se vyskytují na více po sobě jdoucích snímcích. Pokud se tedy vozidlo objevuje na několika po sobě jdoucích snímcích, bere se v potaz pouze nejlepší výsledek z dané množiny snímků.

Tabulka 5.3: Srovnání úspěšnosti klasifikace s využitím metody Unpack proti přímé klasifikaci.

Měření	Přímá 2D	Unpack 3D
Accuracy	75.58%	76.23%
Track Acc.	84.03%	86.06%

5.5 Úspěšnost systému na nezávislých datech

Tato část vyhodnocení je poněkud vágní a slouží spíše pro ověření funkce systému mimo snímky datasetu než poskytnutí přesných statistik. Základem vyhodnocení je vstupní video pořízené kamerou mobilního telefonu s využitím stativu. Záznam byl pořízen během běžných denních světelných podmínek a jeho kvalita odpovídá běžné dohledové kameře. Vstupní video zabírá silnici se 2 pruhy v každém směru tak, že vozidla jedoucí směrem ke kameře jsou lépe viditelná. Ukázka snímků z testovacího videa je součástí přílohy B.

Vzhledem k absenci anotací probíhá vyhodnocení přímým sledováním výstupu systému. Problém nastává především u klasifikace typu vozidel, protože běžný uživatel sám nedokáže správně určit třídu všech vozidel a všechna vozidla ani nejsou součástí klasifikátoru. Zmíněně statistiky jsou vždy vztažené k vozidlům na celé jejich trajektorii, nikoli k jednotlivým snímkům. Pro uznání klasifikace stačí, aby byla třída vozidla správně určena alespoň jedenkrát.

V průběhu záznamu projelo po silnici 60 osobních vozidel z toho 31 směrem ke kameře a 29 směrem od kamery. Všech 60 vozidel bylo úspěšně detekováno a to v zásadě stabilně na celé své dráze. Odhadnuté 3D bounding boxy vozidel odpovídají skutečnosti až na drobné anomálie a odchylky způsobené občasným nepřesným určením kontury. U klasifikace bylo z 60 vozidel nejméně 37 správně rozpoznáno s tím, že mnoho nerozpoznaných zjevně není součástí trénovaného datasetu, nicméně přesný údaj není znám.

Kapitola 6

Závěr

Cílem této práce bylo vytvoření systému pro rozpoznání typu vozidla s využitím dohledové kamery. Navržený systém pracuje s klasifikací typu vozidel pomocí konvolučních neuronových sítí s využitím normalizační metody Unpack. Cílem této metody je převedení obrazu vozidla do jeho zdánlivé rovinné podoby, čímž jsou minimalizovány rozdíly pozorovacích úhlů kamer. Vyhodnocením metody bylo dokázáno, že aplikací metody lze dosáhnout zlepšení výsledků klasifikace. Konkrétně se jednalo o zlepšení absolutní úspěšnosti o 2% oproti metodě přímé klasifikace na výsledných 86% úspěšně klasifikovaných vozidel s využitím architektury sítě InceptionV3. Přesto, že očekávané zlepšení bylo vyšší, lze tento výsledek považovat za uspokojivý a dokazuje prospěšnost normalizace vstupních dat pro účely klasifikace a potenciál dalšího rozvoje metody.

Výsledný systém pracuje s jednotlivými snímky nebo videem z dopravní dohledové kamery bez kladených omezení na úhel pohledu scény a nutnosti jakékoliv individuální konfigurace pro dané podmínky. Funkčnost systému byla také ověřena na nezávislých datech. Tímto byly splněny všechny body dle zadání práce.

Možné pokračování vývoje systému je především realizace sledování trajektorie vozidel v obraze pro jejich zpětnou identifikaci a tím zajištění stabilních výsledků při celém průjezdu vozidla scénou. Mezi jednotlivými snímky videa totiž často dochází k různým nepřesnostem určení kontury, která se přenáší do chybné klasifikace. S využitím sledování vozidel by bylo možné tyto anomálie identifikovat a eliminovat.

Literatura

[1] Brundage, M.; Avin, S.; Clark, J.; aj.: The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation. *ArXiv e-prints*, Únor 2018, 1802.07228.

[2] Canziani, A.; Paszke, A.; Culurciello, E.: An Analysis of Deep Neural Network Models for Practical Applications. *ArXiv e-prints*, Květen 2016, 1605.07678.

[3] Clady, X.; Negri, P.; Milgram, M.; aj.: Multi-class Vehicle Type Recognition System. In *Artificial Neural Networks in Pattern Recognition*, editace L. Prevost; S. Marinai; F. Schwenker, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, ISBN 978-3-540-69939-2, s. 228–239.

[4] Collobert, R.; Puhrsch, C.; Synnaeve, G.: Wav2Letter: an End-to-End ConvNet-based Speech Recognition System. *ArXiv e-prints*, Září 2016, 1609.03193.

[5] D., P.: *A Few Useful Things to Know about Machine Learning*. [Online; navštíveno 24.04.2018].
URL https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf

[6] Fang, J.; Zhou, Y.; Yu, Y.; aj.: Fine-Grained Vehicle Model Recognition Using A Coarse-to-Fine Convolutional Neural Network Architecture. *IEEE Transactions on Intelligent Transportation Systems*, ročník 18, č. 7, July 2017: s. 1782–1792, ISSN 1524-9050, doi:10.1109/TITS.2016.2620495.

[7] G., E.: *Introduction to Supervised Learning*. [Online; navštíveno 25.04.2018].
URL https://people.cs.umass.edu/~elm/Teaching/Docs/supervised2014a.pdf

[8] Girshick, R.; Donahue, J.; Darrell, T.; aj.: Rich feature hierarchies for accurate object detection and semantic segmentation. *ArXiv e-prints*, Listopad 2013, 1311.2524.

[9] Goodfellow, I.; Bengio, Y.; Courville, A.: *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[10] He, K.; Zhang, X.; Ren, S.; aj.: Deep Residual Learning for Image Recognition. *ArXiv e-prints*, Prosinec 2015, 1512.03385.

[11] Jha, G.: Artificial Neural Networks and Its Applications. 04 2018.

[12] Karpathy, A.: *Convolutional Neural Networks (CNNs / ConvNets)*. [Online; navštíveno 24.04.2018].
URL http://cs231n.github.io/convolutional-networks/

[13] Karpathy, A.: *Image Classification*. [Online; navštíveno 24.04.2018].
URL http://cs231n.github.io/classification/

[14] Krause, J.; Stark, M.; Deng, J.; aj.: 3D object representations for fine-grained categorization. In *Proceedings - 2013 IEEE International Conference on Computer Vision Workshops, ICCVW 2013*, United States: Institute of Electrical and Electronics Engineers Inc., 1 2013, ISBN 9781479930227, s. 554–561, doi:10.1109/ICCVW.2013.77.

[15] Nielsen, M. A.: *Neural Networks and Deep Learning*. Determination Press, 2015, http://neuralnetworksanddeeplearning.com/.

[16] Psyllos, A.; Anagnostopoulos, C.; Kayafas, E.: Vehicle model recognition from frontal view image measurements. *Computer Standards & Interfaces*, ročník 33, č. 2, 2011: s. 142 – 151, ISSN 0920-5489, doi:https://doi.org/10.1016/j.csi.2010.06.005, xVI IMEKO TC4 Symposium Exploring New Frontiers of Instrumentation and Methods for Electrical and Electronic Measurements and XIII International Workshop on ADC Modelling and Testing.
URL http://www.sciencedirect.com/science/article/pii/S0920548910000838

[17] Ramnath, K.; Sinha, S. N.; Szeliski, R.; aj.: Car make and model recognition using 3D curve alignment. In *IEEE Winter Conference on Applications of Computer Vision*, March 2014, ISSN 1550-5790, s. 285–292, doi:10.1109/WACV.2014.6836087.

[18] Ren, S.; He, K.; Girshick, R.; aj.: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *ArXiv e-prints*, Červen 2015, 1506.01497.

[19] Roman, J.; Adam, H.; Markéta, D.; aj.: Real-Time Pose Estimation Piggybacked on Object Detection. In *ICCV*, 2015.

[20] Sermanet, P.; Eigen, D.; Zhang, X.; aj.: OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. *ArXiv e-prints*, Prosinec 2013, 1312.6229.

[21] Sochor, J.; Špaňhel, J.; Herout, A.: BoxCars: Improving Fine-Grained Recognition of Vehicles Using 3-D Bounding Boxes in Traffic Surveillance. *IEEE Transactions on Intelligent Transportation Systems*, 2018: s. 1–12, ISSN 1524-9050, doi:10.1109/TITS.2018.2799228.

[22] Szegedy, C.; Vanhoucke, V.; Ioffe, S.; aj.: Rethinking the Inception Architecture for Computer Vision. *ArXiv e-prints*, Prosinec 2015, 1512.00567.

[23] Wu, H.; Gu, X.: Towards Dropout Training for Convolutional Neural Networks. *ArXiv e-prints*, Prosinec 2015, 1512.00242.

[24] Yang, J.; Price, B.; Cohen, S.; aj.: Object Contour Detection with a Fully Convolutional Encoder-Decoder Network. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, ročník 2016-January, United States: IEEE Computer Society, s. 193–202.

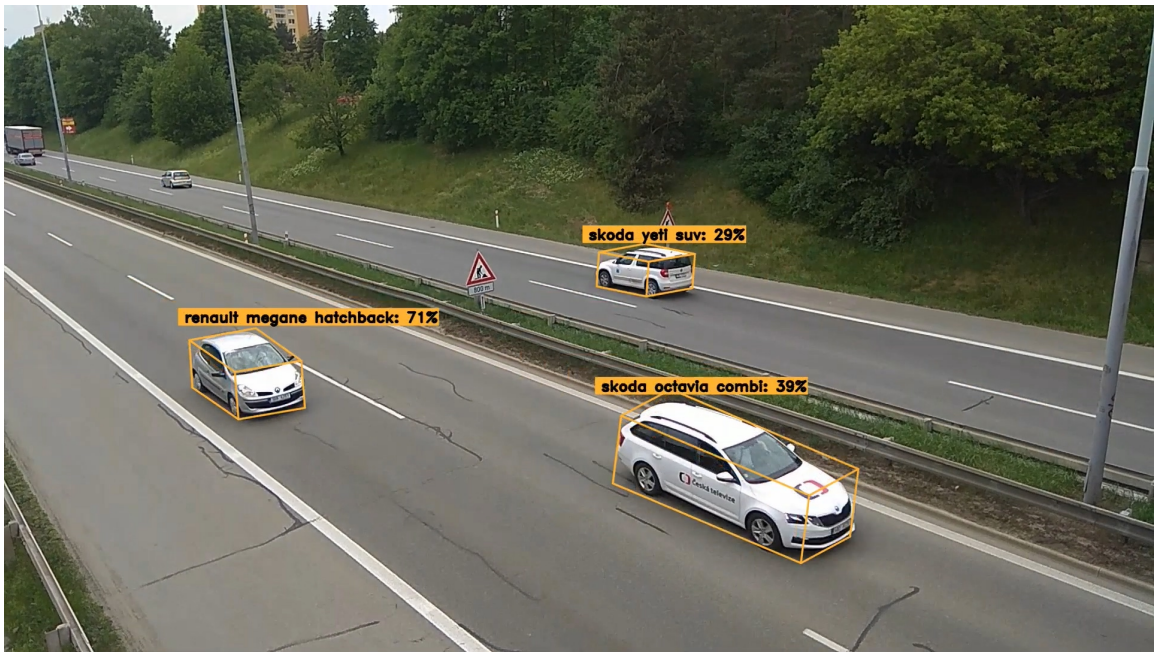
Příloha A

Konfigurační soubor

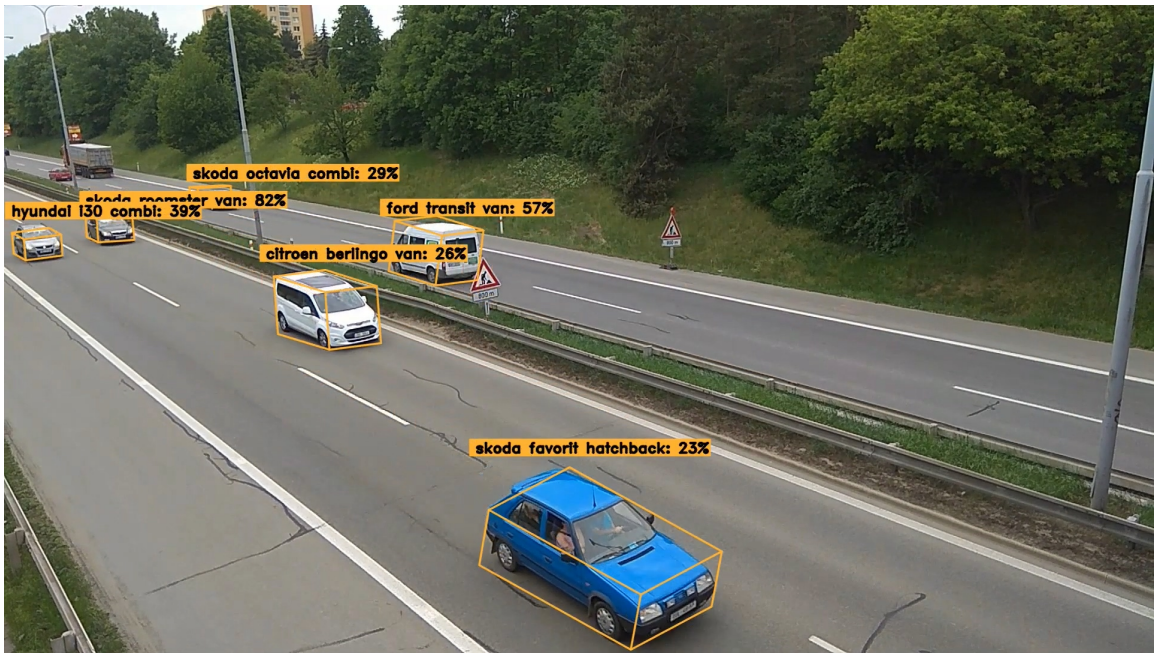
- `INPUT_PATH` – adresář se vstupními daty
- `OUTPUT_PATH` – adresář pro výstup výsledných dat
- `MODELS_PATH` – adresář s natrénovanými modely pro klasifikaci
- `DETECTION_MODEL` – model pro detekci vozidel
- `DETECTION_MIN_CONFIDENCE` – minimální hodnota věrohodnosti pro uznání detekce
- `ANGLE_ESTIM_MODEL` – model pro odhad směru k úběžníkům
- `CONTOUR_ESTIM_MODEL` – model pro detekci kontury
- `CLASSIFICATION_MODEL` – model pro klasifikaci typu vozidel s využitím Unpack
- `CLASSIFICATION_SPLIT` – binární soubor s mapováním tříd vozidel na jejich názvy
- `SPLIT_NAME` – použité mapování tříd vozidel
- `CLASSIFICATION_MIN_CONFIDENCE` – minimální hodnota věrohodnosti pro uznání klasifikace
- `OUTPUT_TO_FILE` – povolení výstupu obrazových dat do souboru
- `OUTPUT_TO_WINDOW` – povolení výstupu obrazových dat do náhledového okna
- `ANNOT_BB2D` – povolení vykreslení detekovaných regionů
- `ANNOT_BB3D` – povolení vykreslení získaných 3D bounding boxů
- `ANNOT_ANGLES` – povolení vykreslení získaných směrů k úběžníkům
- `ANNOT_CONTOUR` – povolení vykreslení získaných kontur vozidel
- `ANNOT_CLASSES` – povolení vykreslení názvů třídy vozidel
- `SAVE_PASSAGE_PICKLE` – povolení uložení binárních souborů všech zpracovaných dat (pro ladící účely)

Příloha B

Ukázka výstupu systému



Obrázek B.1: Ukázka výstupu systému.



Obrázek B.2: Ukázka výstupu systému.

Obrázek B.3: Ukázka výstupu systému.